I590 – Python Spring 2020

Assignment No:	9	Issued:	March 8
Title:	Data Processing with NumPy	Due:	March 15

Notes:

- Use *NumPy* to handle data in tabular form.
- Wherever possible use vectorized *NumPy* operations (instead of loops).

Data

Final grades in Computer Science CS333 course are based on the weights of the assignment groups:

Labs	30 %
Homework	20 %
Quiz	15 %
Midterm 1	10 %
Midterm 2	15 %
Final project	10 %
Total	100 %

Midterm 1, *Midterm 2* and *Final project* are assignment groups with a single assignment. Other assignment groups contain several assignments that together constitute the grade for the group. For example, there are 6 homework assignments that together contribute 20% towards the final grade.

Excel file "Scores_all.xlsx" contains all grades at the end of the semester. Column *A* contains the student ID numbers and the other columns contain the grades:

A	Student ID numbers
B to G:	Homework
H to R:	Labs
S:	Final project
T:	Midterm 1
U to AC:	Quizzes (Note that Quiz 4 is missing)
AD:	Midterm 2

Rows contain the following information:

1:	Headings (textual description of columns)
2:	Weights of individual assignment within its group
3 to 46:	Grades of students in corresponding assignments

Homework and lab assignments, final project and midterms are graded from 0 to 100 points. The quizzes are graded from 0 to 10 points.

Question 1 (20 points)

Open file "Scores_all.xlsx" with Excel and save it in plain text using "csv" format (Comma Separated Values) to file "Scores_all.csv".

Write a Python program that uses *NumPy* to:

- Read "Scores_all.csv",
- Delete the first row (with column headings),
- Fill out the missing data with 0s,
- Print the resulting data array and save it to the file "scores.npy".

Hints

Open file "Scores_all.xlsx" with Excel and inspect the data.

Note the following:

- First row contains the column headings. Second row contains the weights for the individual assignments.
- The first column contains student ID numbers.
- There are 6 homework assignments (*HW 0* to *HW 5*), 11 labs (*Lab 01* to *Lab 11*), and 9 quizzes (*Quiz 01* to *Quiz 10*). Note that there is no *Quiz 4,* due to an error it was dropped from the scores.
- Some scores are missing, for example, student with *ID = 38180* missed first two homework assignments. The missing scores should be filled out with *0s*.

Use Excel option:

```
File -> Save as -> CSV (Comma delimited)
```

to save the data to file "Scores_all.csv". You will need function `genfromtxt()` to read `.csv` file:

```
from   numpy import genfromtxt
```

Read file "Scores_all.csv" into an *np* array "scores_all":

```
scores_all = genfromtxt("Scores_all.csv", delimiter = ",")
```

You will notice that the empty cells and cells with textual data are read as *NaN* (*Not a Number*). Create a view *"scores"* to "scores_all" that does not contain the first row (with headings). Use statement *np.isnan(scores)* to identify the elements with *NaN* and replace them with *0s*.

Convert the data type of *scores* to integers, print the resulting array *scores* and save it to file "scores.npy".

Sample printout:

```
[[    0   100   100 ...,    10    10   100]
 [49647    90   100 ...,     2     6    89]
 [49865   100    70 ...,     1     7    73]
 ...,
 [29288   100    90 ...,    10    10    74]
 [34759   100    90 ...,     8     9    94]
 [45920    85    85 ...,     0    10    91]]
```

Question 2 (20 points)

Write a Python program that uses *NumPy* to read file "scores.npy" and create a NumPy array "score_headings" with the following columns:

- ID numbers,
- Weighted scores for the homework assignment,
- Weighted scores for the labs,
- Weighted final project scores,
- Weighted midterm 1 scores,
- Weighted scores for the quizzes,
- Weighted midterm 2 scores.

Print "score_headings" and store it to file "score_heading.npy".

Hints

For a student in the row *i*, the weighted score for the homework assignments is:

```
((hw0 + hw1 +hw2 +hw3 + hw4 + hw5)/ 600 ) * 20
```

Instead of using a loop to calculate weights for each student, you are supposed to use vectorized operations. For example, the following statement sums the homework scores for all students:

```
(scores_only[:, 1:7]).sum(axis = 1)
```

all rows — columns 1 to 6 — sum elements in rows

Note that "scores_only" is a view to "scores" without the first row with the weights.

Statement:

```
np.set_printoptions(suppress=True)
```

suppresses the scientific printout of the elements of "score_headings".

Save "score_headings" to file "score_headings.npy" and print with elements rounded to 1 decimal digit:

```
np.around(score_headings, decimals = 1)
```

Sample printout

```
[[ 49647.      17.1     29.3      8.8      5.7      8.2     13.4]
 [ 49865.      18.9     28.4     10.       9.3      9.5     11. ]
 [ 88260.      20.      27.8      9.5      4.4      6.7     12. ]
 [ 73216.      18.7     29.3      9.       7.5      9.8      9.8]
 [ 62539.      18.5     29.       8.8      5.9      9.5      7.4]
 . . .
```

Question 3 (20 points)

Write a function “to_letter_grade(score)” that returns the letter grade for a scalar input “score”. See Question 1, Assignment 6, for details.

Function “main()” reads file “score_headings.npy”, calculates and prints the letter grade for each student.

You are supposed to use NumPy vectorized operations instead of loops.

Hints

Create a NumPy array “final_scores” with a row for each student and 3 columns:

- ID numbers
- Final scores
- Letter grade

Final score for each student is the sum of weighted assignment groups, which are columns 1 to 6 of the “final_scores”. For example, the final score of the student with ID number 49647 is:

```
17.1 + 29.3 + 8.8 + 5.7 + 8.2 + 13.4
```

Round the final scores to 1 decimal digit.

Use function “to_letter_grade(score)” to convert the final scores to letter grades. You will have to “vectorize” the function to be able to apply it to the final scores column:

```
np.vectorize(to_letter_grade)
```

Convert “final_scores” to strings:

```
final_scores = final_scores.astype(str)
```

Insert the letter grades (which are strings) into the third column of “final_scores”.

Sample output

```
[['49647.0' '82.3' 'B-']
 ['49865.0' '87.1' 'B+']
 ['88260.0' '80.4' 'B-']
 ['73216.0' '84.0' 'B']
 ['62539.0' '79.1' 'C+']
 ['38180.0' '70.4' 'C-']
 ['75682.0' '87.4' 'B+']
  . . .
```

Question 4 (20 points)

Write a Python program that uses *NumPy* to calculate and print average score for each assignment.

Hints

Read file “scores.npy”. The columns contain the scores of individual assignment.

An (arithmetic) average is the sum of a series of numbers divided by the count of that series of numbers.

Calculate the average of the columns; make sure that you do not sum the weights of the assignments. Do not use loops, use NumPy vectorized operations to calculate the averages. Please don’t use function ‘mean’, rather use ‘sum’ and calculate the average.

Use loops to print the results in each assignment group rounded to 1 decimal digit.

Sample output

```
Homework 0 average:  80.0
Homework 1 average:  88.2
Homework 2 average:  96.4
Homework 3 average:  89.9
Homework 4 average:  91.0
Homework 5 average:  87.7

Lab      1 average:  98.7
Lab      2 average:  95.1
Lab      3 average:  97.6
Lab      4 average:  100.0
Lab      5 average:  94.1
Lab      6 average:  96.0
Lab      7 average:  89.5
Lab      8 average:  93.9
Lab      9 average:  83.5
Lab      10 average:  91.8
Lab      11 average:  93.9

Quiz     1 average:  10.0
Quiz     2 average:  5.2
Quiz     3 average:  5.1
Quiz     5 average:  7.0
Quiz     6 average:  7.6
Quiz     7 average:  3.4
Quiz     8 average:  5.5
Quiz     9 average:  5.2
Quiz     10 average:  5.6

Final project avg:   77.2
Midterm 1 average:   69.6
Midterm 2 average:   79.4
```

Question 5 (20 points)

Write a Python program that uses *NumPy* to calculate and print average of each assignment group.

Hints

Read file “scores.npy”.

Homework assignment group consists of 6 homework assignments. Calculate the average by summing up all homework scores and dividing by the number of homework submissions (homework elements in “scores.npy”). Repeat the same for the labs and quizzes.

The averages for the assignment groups “Final project”, “Midterm 1” and “Midterm 2” are just the averages of the corresponding columns. Print the averages rounded to 1 decimal digit.

Please don’t use function ‘mean’, rather use ‘sum’ and calculate the average.

Sample output

```
Program to calculate assignment group averages.


Homework average:  88.9
Lab average:       94.0
Quiz average:       6.1
Project average:   77.2
Midterm 1 average: 69.6
Midterm 2 average: 79.4
```

Instructions

- Write your programs in Python 3. Write a separate program for each question.
- Make sure your code is properly formatted, structured and commented. Include a header to program file with your name, date, question number and short program description.
- Do your best to follow the programming style suggested in *Python Enhancement Proposal PEP8* (outlined in a tutorial https://realpython.com/python-pep8/). Follow the naming conventions for variables and functions.
- Make sure that your program compiles. A program that doesn’t compile is usually scored 0 points.

- Each program should be stored in a separate Python file. Please name your files such that it starts with the word "question" and ends with the number of the question. All files should be executable python files (they should all end in .py). Acceptable examples include *Question_5.py*, *question_4.py*, *Question3.py*, and *question2.py.*
- A program should run the entire code for a particular question by directly executing it. Include call to `main()` function at the end of your program. Provide any data files that your program needs to execute.
- Please upload to Canvas separate files for each question. If there are five questions, you will be uploading five separate files.
- If you want to provide more instructions for executing your code, upload a 'readme.txt' file, where you may provide additional information.